Generating Energy-optimized Adaptive Software on a Heterogeneous MPSoC with PREESM

Florian Arrestier*, Karol Desnos*, Julien Heulot*, Alexandre Honorat*, Daniel Ménard*,
Antoine Morvan*, Jean-François Nezan* Maxime Pelcat*†, Claudio Rubattu*‡,
*IETR UMR CNRS 6164, INSA Rennes, Rennes, France
†Institut Pascal, UMR CNRS 6602, Université Clermont Auvergne, Aubière, France
‡Department of Chemistry and Pharmacy, University of Sassari, Sassari, Italy
{first}.{last}@insa-rennes.fr

***Abstract*—This Booth demonstrates how PREESM and SPIDER tools generate energy-optimized sensor-based adaptive software on a heterogeneous platform. PREESM is a rapid system prototyping tool provided with a runtime manager named SPIDER . PREESM simulates stream processing applications and generates code for multi/many-cores. Processing can either be statically mapped or adaptively managed by SPIDER . Steps when using PREESM are: (1) Model your Application: PREESM provides you with a dataflow language, designed to express parallelism. (2) Model your architecture: PREESM simulates and generates code for a wide range of systems (e.g., ARM, DSP, FPGA). (3) Prototype and Run your Design: PREESM takes mapping decisions and provides early design space information such as scheduling, memory use, and core loads. PREESM and SPIDER are available on GitHub, and supported by tutorials and a reactive community. PREESM and SPIDER are part of the H2020 CERBERO framework.**

***Index Terms*—MPSoC programming, Design space exploration, Dataflow models of computation, Adaptive runtime**

I. PREESM FOR DESIGN SPACE EXPLORATION

PREESM [1] is an Eclipse-based environment for parallel applications development with design-time prediction, as well as code generation and code re-use capabilities. PREESM input models are the Parameterized and Interfaced Synchronous Dataflow (PiSDF) [2] for application, and a Model of Architecture (MoA) [3] for platform. PREESM simulates the execution and provides early performance predictions in terms of latency, workload, and memory footprint. Moreover, it generates a time and memory optimized code to execute the application on a range of parallel and heterogeneous architectures.

II. SPIDER FOR RUNTIME APPLICATION MANAGEMENT

Many applications have data dependent computational loads. Their resource requirements and parallelism depend on constantly changing parameters acquired from sensors or from user commands. On platform architecture side, the availability of resources such as cores, accelerators or subsystems may change at runtime, due to either subsystem failures or dynamic hardware reconfiguration. SPIDER is a system runtime manager that takes on-the-fly resource re-mapping decisions based on these modifications.

Authors are listed in alphabetical order. The presented work has received funding from the European Unions Horizon 2020 research and innovation programme under grant agreement No 732105.

TABLE I
ENERGY CONSUMPTION OF AN IMAGE FILTERING MANAGED BY SPIDER ON AN 8-CORE HETEROGENEOUS PROCESSOR.

Algorithm	Energy with platform heterogeneity information	Energy w/o platform heterogeneity information
Full filter	2.35 J/frame	>4 J/frame
Reduced filter	1.92 J/frame	2.3 J/frame

SPIDER and PREESM are strongly coupled and rely on the same models of computation and architecture that balance expressiveness and system predictability. Models are constantly updated to match system state. SPIDER is implemented as a C++ library for scheduling applications at runtime on multi/many-core/accelerated heterogeneous systems. SPIDER is composed of a global runtime that takes re-mapping decisions and issues job execution requests to local runtimes managing the execution of self-contained tasks. The main performance indicator observed in SPIDER is latency.

III. GENERATING ENERGY-OPTIMIZED MPSOC SOFTWARE

Table I shows experimental results[1] on the energy consumption measured per frame for an image filter running on an 8-core ARM big.LITTLE architecture managed by SPIDER . Two SPIDER cases are considered: with or without platform heterogeneity knowledge. SPIDER can also reconfigure between two application configurations. Having a knowledge on the platform heterogeneity during re-mapping is demonstrated on the example to save between 16% and 40% of energy, while SPIDER manages to reduce energy when filtering effort is reduced. Code, tutorials and documentation are on preesm.org.

REFERENCES

[1] M. Pelcat, K. Desnos, J. Heulot, C. Guy, J.-F. Nezan, and S. Aridhi, "PREESM: A dataflow-based rapid prototyping framework for simplifying multicore DSP programming," in *EDERC Conference*, 2014.
[2] K. Desnos, M. Pelcat, J.-F. Nezan, S. S. Bhattacharyya, and S. Aridhi, "Pimm: Parameterized and interfaced dataflow meta-model for MPSoCs runtime reconfiguration," in *SAMOS XIII*, 2013.
[3] M. Pelcat, A. Mercat, K. Desnos, L. Maggiani, Y. Liu, J. Heulot, J. F. Nezan, W. Hamidouche, D. Menard, and S. S. Bhattacharyya, "Reproducible evaluation of system efficiency with a model of architecture: From theory to practice," *IEEE TCAD*, 2017.

[1] Check experiments in: http://youtu.be/a9WIucWfjkU.